Departamento de Lenguajes y Sistemas Informáticos e Ingeniería del Software UPM ETSIINF.

Examen de Programación II. Convocatoria ordinaria. 31-05-2017.

Realización: *El test se realizará en la hoja de respuesta. Es* ***importante*** *que no olvidéis rellenar vuestros datos personales y el código clave de vuestro enunciado. Se pueden utilizar hojas aparte en sucio.*

Duración: *La duración total del test será de* ***30 minutos****.*

Peso en el examen: *El examen tendrá dos partes, un test y un ejercicio práctico. El test tendrá un peso de un* ***30 %*** *en la nota del examen y el ejercicio práctico tendrá el* ***70 %*** *restante.*

Puntuación: *El test se valora sobre* ***10 puntos****. Las preguntas tipo test pueden tener una única respuesta o varias respuestas, el enunciado lo deja claro. Cada pregunta con una única respuesta respondida correctamente vale 2 puntos, e incorrectamente respondida resta 2/3 puntos. Si en una pregunta con una única respuesta se selecciona más de una respuesta, la pregunta se puntuará con 0 puntos. Para una pregunta con varias respuestas, cada afirmación correcta seleccionada suma* $2/no\_respuestas\_correctas$ *puntos, y cada afirmación incorrecta seleccionada resta* $2/no\_respuestas\_incorrectas$ *puntos. Las preguntas no contestadas suman 0 puntos en cualquier caso.*

Calificaciones: *Las calificaciones se publicarán en moodle como muy tarde el día* ***5 de junio de 2017***

Revisión: *Las revisiones serán el día* ***8 de junio de 2017*** *previa petición por correo electrónico al profesor que se indique por el foro de la asignatura.*

Primer Ejercicio

Dado el siguiente código, y suponiendo que en el mismo paquete ya existe la clase NumeroNegativoExp:

```
public class EjemploFactorial {

  public int factorial(int num) {
    if (num == 0) {
        return 1;
    } else {
        return num * factorial(num-1);
    }
  }

  public int factorialExcepcion(int num)
     throws NumeroNegativoExp {
    if (num < 0) {
        throw new NumeroNegativoExp();
    }
    return factorial(num);
  }

  public static void main (String[] args) {
    EjemploFactorial ejemplo =
        new EjemploFactorial();

    try {
      System.out.print(
        ejemplo.factorialExcepcion(-1));
      System.out.print("Se produjo un error ");
    }
    catch (NumeroNegativoExp e) {
      System.out.print("Terminó con error ");
      System.out.print(ejemplo.factorial(3));
    }
  }
}
```

Pregunta (2 puntos)

Indicar cuál será la salida por consola al ejecutar el programa principal `main`. **Sólo una respuesta es correcta**

a) Terminó con error 6

b) Terminó con error

c) Se produjo un error Terminó con error 6

d) Se produjo un error

Segundo Ejercicio

Dado el siguiente código que utiliza la clase Node<E> vista en clase:

```
public static void main(String[] args) {
  Node<Integer> head, aux;
  head = new Node<Integer>(3, null);
  head.setNext(new Node<Integer>(1, null));
  aux = head;
  head = new Node<Integer>(9, aux);

  while(head != null){
    System.out.print(head.element() + " ");
    head = head.next();
  }
}
```

Pregunta (2 puntos)

Indicar cuál será la salida por consola al ejecutar el programa principal `main`. **Sólo una respuesta es correcta**

a) 9 3 1

b) 9 1 3

c) 3 1 9

d) 3 1

Tercer Ejercicio

Dadas las siguientes afirmaciones sobre interfaces en Java.

Pregunta (2 puntos)

Señala todas las afirmaciones correctas. **Puede haber más de una afirmación correcta**

a) Si una clase A implementa una interfaz I, NO se pueden crear objetos de tipo A.

b) No puede haber más de una clase que implemente una misma interfaz en el mismo programa.

c) Una interfaz permite definir constantes.

d) Una interfaz se puede utilizar para especificar un TAD.

Continúa...

Cuarto Ejercicio

Dadas las siguientes afirmaciones sobre herencia de clases en Java.

Pregunta (2 puntos)

Señala todas las afirmaciones correctas. **Puede haber más de una afirmación correcta**

a) Una clase hija incluye todos los atributos de su clase padre.

b) Java permite herencia múltiple (una clase hija puede tener varias clases padre).

c) Todas las clases heredan o son subclases de la clase Object.

d) Una clase hija solo puede sobreescribir los métodos abstractos que herede de la clase padre.

Quinto Ejercicio

Dada la siguiente clase NavegadorWeb y suponiendo que está definida la excepción ExcepcionNoHayAnteriores:

```
public class NavegadorWeb {
  private String webAbierta;
  private Stack<String> anteriores;
  private Stack<String> posteriores;

  public NavegadorWeb(){
    anteriores = new Stack<String>();
    posteriores = new Stack<String>();
    webAbierta = "página vacía";
  }

  public void openWeb(String url){
    anteriores.push(webAbierta);
    webAbierta = url;
  }

  public void retroceder()
     throws ExcepcionNoHayAnteriores {
    String webAnterior = webAbierta;
    try {
        webAbierta = anteriores.pop();
        posteriores.push(webAnterior);
    } catch (EmptyStackException e) {
        throw new ExcepcionNoHayAnteriores();
    }
  }

  public String toString(){
    return webAbierta;
  }
}
```

Pregunta (2 puntos)

Indica cuál es la salida por consola que muestra la ejecución del siguiente código:

```
public static void main(String[] args) {
  NavegadorWeb webBrowser =
        new NavegadorWeb();

  webBrowser.openWeb("www.a.com");
  webBrowser.openWeb("www.b.com");
  try {
    webBrowser.retroceder();
    System.out.print(webBrowser);
    webBrowser.openWeb("www.c.com");
    webBrowser.retroceder();
    webBrowser.retroceder();
    System.out.print(" " + webBrowser);
  }
  catch (ExcepcionNoHayAnteriores e) {
    System.out.print("error");
  }
}
```

Sólo una respuesta es correcta.

a) www.b.com página vacía

b) www.a.com página vacía

c) www.a.com www.a.com

d) www.a.com error

Fin.